

5

**SYSTEM FOR GENERATING OPTIMIZED COMPUTER DATA FIELD
CONVERSION ROUTINES**

FIELD OF THE INVENTION

The present invention is directed to computer data. More particularly, the present invention is directed to the conversion of one type of computer data field to another type.

BACKGROUND OF THE INVENTION

15 In many instances during computer processing of information, computer data must be converted from one data field type to another. For example, whenever data is passed from one program to another, the data typically goes through several conversions during the process, such as converting from text digits to a binary number.

5

The typical technique for converting data includes using a generic data conversion routine. When an entire record of data must be converted, the conversion routine must determine what the characteristics or attributes are for each of the data fields in the record. This may require the conversion routine to execute the same decision tree for each field for each record even though each field has known characteristics that do not change on a row by row basis. Therefore, many computer cycles are wasted by asking questions such as "Is this field of type character, integer, etc.?" over and over for each data field.

Based on the foregoing, there is a need for a system that provides efficient conversion of data fields.

SUMMARY OF THE INVENTION

15

One embodiment of the present invention is a system for converting data from input field types to output field types. The system receives a plurality of input attributes and output attributes from an application program, dynamically generates a plurality of data field conversion routines for each set of input attributes and output attributes, and stores the plurality of data field conversion routines in memory that is accessible to the application program.

BRIEF DESCRIPTION OF THE DRAWINGS

Fig. 1 is a block diagram that illustrates an overview of the functionality of an optimized conversion generator system in accordance with one embodiment of the present invention.

5 Fig. 2 is a flowchart of the steps performed by the system in accordance with one embodiment of the present invention to generate optimized conversion routines.

Fig. 3 is a flowchart of the steps executed by the application when using the routines to convert input fields to output fields.

Fig. 4 is a flowchart of the code generating steps executed by the conversion generator system to generate code when called by the application.

Figs. 5a and 5b illustrate a general example of dynamic code building that is used in one embodiment of the present invention.

15 Figs. 6a - 6h illustrate a specific example of a dynamic code generation routine that performs CHARACTER to CHARACTER conversions.

DETAILED DESCRIPTION

One embodiment of the present invention is a system that generates optimized data field to data field conversion routines for each type of conversion required by an application program. Fig. 1 is a block diagram that illustrates an overview of the functionality of an optimized conversion generator system 20 in accordance with one

embodiment of the present invention. System 20 can be implemented in software and executed on a general purpose computer that includes a central processing unit, and memory. In one embodiment, system 20 is implemented with IBM/360 machine instructions.

5 An application program 10 requires one or more types of field conversions to be executed. For each type of conversion, application 10 provides to system 20 the input (or "source") and output (or "destination") field attributes. For each set of input and output field attributes, system 20 dynamically generates an optimized conversion routine 30 that performs the conversion. The optimized routines 30 are placed in storage that is available to application 10.

10 15 20 25 30 35 40 45 50 55 60 65 70 75 80 85 90 95

The routines 30 in one embodiment are generated as stand-alone routines that are capable of being serially reusable and are called by application 10 using, for example, an application program interface ("API") when a conversion is required. In another embodiment, the routines 30 are generated as code chunks that are inserted inline with application 10 and are directly accessed when a conversion is required.

15 One benefit of the present invention is that by building optimized conversion routines specifically tailored to the input and output field attributes, every execution of the routine saves numerous instructions that would normally be needed to identify field attributes each time the conversion is executed.

Fig. 2 is a flowchart of the steps performed by system 20 in accordance with one embodiment of the present invention to generate optimized conversion routines 30. The steps are executed by system 20 after application 10 determines at step 100 what attributes the input fields and output fields have.

5 At step 102, system 20 sets up the default process options of the generated conversion routines 30. The options may include whether the generated conversion routines 30 will be callable functions (i.e., able to be called by application 10), or copied inline into application 10. Step 102 builds a template interface block 104 which is an interface between application 10 and conversion generator system 20. Step 102 also generates an initiation call 106 that obtains the necessary storage and checks for errors.

At step 108, a loop is initiated. The loop continues until all fields that must be converted are exhausted.

15 Within the loop, at step 110 each set of input and output field attributes is received from application 10. The attributes are received through an API, and step 110 also builds a common field conversion interface block 116 based on the attributes.

At step 112, the code generator of system 20 is called, using the common interface block 116. Step 112 generates code 118.

20 At step 114, a function pointer that points to the generated field conversion routine 30 is saved.

Fig. 3 is a flowchart of the steps executed by application 10 when using routines 30 to convert input fields to output fields.

During step 122, the application is processing. At step 124, the application obtains source or input data to convert. Typically, step 124 involves reading one or 5 more records.

At step 126, a loop is initiated for each record read. At step 128, in one embodiment the appropriate conversion routine 30 for the conversion is called.

When all the data field and records are converted, at step 132 the code generator system 20 is called for termination. This results in freeing up memory at step 134.

At step 136, application 10 continues to process. Finally, at step 138 application 10 is completed.

Fig. 4 is a flowchart of the code generating steps executed by conversion generator system 20 to generate code when called by application 10.

At step 200, system 20 initializes by, for example, establishing the required storage, checking for invalid options, and specifying how the code should be generated.

At step 202, system 20 validates specific field conversion options such as verifying that the input and output lengths are correct. Step 202 also determines how

big the code will be when generated. This can be used by application 10 if the generated code will be stored inline.

At step 204, system 20 builds the conversion routine using field conversion interface block 116.

5 At step 206, the storage obtained at step 200 is released.

Steps 202 and 204 go through the same internal process. Therefore, at step 208 the input field type is determined. Examples of input field types include character input 210 or special time format input 212. However, any input field type is supported by the present invention.

Similarly, at step 214 the output field type is determined. Examples of output field types also include character input 213 or special time format input 215, but any output field type is supported by the present invention.

At step 216, if step 202 was executed, the size of the generated code is determined. At step 218, if step 204 was executed, the field conversion routines 30 are generated.

15 As disclosed, system 20 in accordance with one embodiment of the present invention dynamically generates optimized conversion routines 30 for each set of input and output field attributes. Routines 30 are then utilized by application 10 to process conversions. Input and output fields are categorized into archetypal data types by system 20, each with definable attributes and conversion behaviors. For example:

- Character data types will be a fixed length field with a maximum length attribute and a CCSID (or character set code page) attribute.
- Date data types will be a fixed length field with a maximum length attribute and a format attribute (ISO, EUR, etc) which determines location and type of separators used in date.

5

Some previously described or additional features included in one embodiment of optimized conversion generator system 20 include:

- Optionally obtain and free storage for API control blocks and/or generated code.
- API control blocks can be chained and templated by API management functions.
- API control blocks can be built through use of a macro interface.
- Conversion routines can utilize registers to address the input and output field locations directly. The registers can be chosen by application 10 through API parameters.
- The source field address register may optionally be incremented to the end of the input field after conversion based on API parameters.
- The target field address register may optionally be incremented to the end of the formatted field after conversion based on API parameters.

15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
3310
3311
3312
3313
3314
3315
3316
3317
3318
3319
3320
3321
3322
3323
3324
3325
3326
3327
3328
3329
3330
3331
3332
3333
3334
3335
3336
3337
3338
3339
33310
33311
33312
33313
33314
33315
33316
33317
33318
33319
33320
33321
33322
33323
33324
33325
33326
33327
33328
33329
33330
33331
33332
33333
33334
33335
33336
33337
33338
33339
333310
333311
333312
333313
333314
333315
333316
333317
333318
333319
333320
333321
333322
333323
333324
333325
333326
333327
333328
333329
333330
333331
333332
333333
333334
333335
333336
333337
333338
333339
3333310
3333311
3333312
3333313
3333314
3333315
3333316
3333317
3333318
3333319
3333320
3333321
3333322
3333323
3333324
3333325
3333326
3333327
3333328
3333329
3333330
3333331
3333332
3333333
3333334
3333335
3333336
3333337
3333338
3333339
33333310
33333311
33333312
33333313
33333314
33333315
33333316
33333317
33333318
33333319
33333320
33333321
33333322
33333323
33333324
33333325
33333326
33333327
33333328
33333329
33333330
33333331
33333332
33333333
33333334
33333335
33333336
33333337
33333338
33333339
333333310
333333311
333333312
333333313
333333314
333333315
333333316
333333317
333333318
333333319
333333320
333333321
333333322
333333323
333333324
333333325
333333326
333333327
333333328
333333329
333333330
333333331
333333332
333333333
333333334
333333335
333333336
333333337
333333338
333333339
3333333310
3333333311
3333333312
3333333313
3333333314
3333333315
3333333316
3333333317
3333333318
3333333319
3333333320
3333333321
3333333322
3333333323
3333333324
3333333325
3333333326
3333333327
3333333328
3333333329
3333333330
3333333331
3333333332
3333333333
3333333334
3333333335
3333333336
3333333337
3333333338
3333333339
33333333310
33333333311
33333333312
33333333313
33333333314
33333333315
33333333316
33333333317
33333333318
33333333319
33333333320
33333333321
33333333322
33333333323
33333333324
33333333325
33333333326
33333333327
33333333328
33333333329
33333333330
33333333331
33333333332
33333333333
33333333334
33333333335
33333333336
33333333337
33333333338
33333333339
333333333310
333333333311
333333333312
333333333313
333333333314
333333333315
333333333316
333333333317
333333333318
333333333319
333333333320
333333333321
333333333322
333333333323
333333333324
333333333325
333333333326
333333333327
333333333328
333333333329
333333333330
333333333331
333333333332
333333333333
333333333334
333333333335
333333333336
333333333337
333333333338
333333333339
3333333333310
3333333333311
3333333333312
3333333333313
3333333333314
3333333333315
3333333333316
3333333333317
3333333333318
3333333333319
3333333333320
3333333333321
3333333333322
3333333333323
3333333333324
3333333333325
3333333333326
3333333333327
3333333333328
3333333333329
3333333333330
3333333333331
3333333333332
3333333333333
3333333333334
3333333333335
3333333333336
3333333333337
3333333333338
3333333333339
33333333333310
33333333333311
33333333333312
33333333333313
33333333333314
33333333333315
33333333333316
33333333333317
33333333333318
33333333333319
33333333333320
33333333333321
33333333333322
33333333333323
33333333333324
33333333333325
33333333333326
33333333333327
33333333333328
33333333333329
33333333333330
33333333333331
33333333333332
33333333333333
33333333333334
33333333333335
33333333333336
33333333333337
33333333333338
33333333333339
333333333333310
333333333333311
333333333333312
333333333333313
333333333333314
333333333333315
333333333333316
333333333333317
333333333333318
333333333333319
333333333333320
333333333333321
333333333333322
333333333333323
333333333333324
333333333333325
333333333333326
333333333333327
333333333333328
333333333333329
333333333333330
333333333333331
333333333333332
333333333333333
333333333333334
333333333333335
333333333333336
333333333333337
333333333333338
333333333333339
3333333333333310
3333333333333311
3333333333333312
3333333333333313
3333333333333314
3333333333333315
3333333333333316
3333333333333317
3333333333333318
3333333333333319
3333333333333320
3333333333333321
3333333333333322
3333333333333323
3333333333333324
3333333333333325
3333333333333326
3333333333333327
3333333333333328
3333333333333329
3333333333333330
3333333333333331
3333333333333332
3333333333333333
3333333333333334
3333333333333335
3333333333333336
3333333333333337
3333333333333338
3333333333333339
33333333333333310
33333333333333311
33333333333333312
33333333333333313
33333333333333314
33333333333333315
33333333333333316
33333333333333317
33333333333333318
33333333333333319
33333333333333320
33333333333333321
33333333333333322
33333333333333323
33333333333333324
33333333333333325
33333333333333326
33333333333333327
33333333333333328
33333333333333329
33333333333333330
33333333333333331
33333333333333332
33333333333333333
33333333333333334
33333333333333335
33333333333333336
33333333333333337
33333333333333338
33333333333333339
333333333333333310
333333333333333311
333333333333333312
333333333333333313
333333333333333314
333333333333333315
333333333333333316
333333333333333317
333333333333333318
333333333333333319
333333333333333320
333333333333333321
333333333333333322
333333333333333323
333333333333333324
333333333333333325
333333333333333326
333333333333333327
333333333333333328
333333333333333329
333333333333333330
333333333333333331
333333333333333332
333333333333333333
333333333333333334
333333333333333335
333333333333333336
333333333333333337
333333333333333338
333333333333333339
3333333333333333310
3333333333333333311
3333333333333333312
3333333333333333313
3333333333333333314
3333333333333333315
3333333333333333316
3333333333333333317
3333333333333333318
3333333333333333319
3333333333333333320
3333333333333333321
3333333333333333322
3333333333333333323
3333333333333333324
3333333333333333325
3333333333333333326
3333333333333333327
3333333333333333328
3333333333333333329
3333333333333333330
3333333333333333331
3333333333333333332
3333333333333333333
3333333333333333334
3333333333333333335
3333333333333333336
3333333333333333337
3333333333333333338
3333333333333333339
33333333333333333310
33333333333333333311
33333333333333333312
33333333333333333313
33333333333333333314
33333333333333333315
33333333333333333316
33333333333333333317
33333333333333333318
33333333333333333319
33333333333333333320
33333333333333333321
33333333333333333322
33333333333333333323
33333333333333333324
33333333333333333325
33333333333333333326
33333333333333333327
33333333333333333328
33333333333333333329
33333333333333333330
33333333333333333331
33333333333333333332
33333333333333333333
33333333333333333334
33333333333333333335
33333333333333333336
33333333333333333337
33333333333333333338
33333333333333333339
333333333333333333310
333333333333333333311
333333333333333333312
333333333333333333313
333333333333333333314
333333333333333333315
333333333333333333316
333333333333333333317
333333333333333333318
333333333333333333319
333333333333333333320
333333333333333333321
333333333333333333322
333333333333333333323
333333333333333333324
333333333333333333325
333333333333333333326
333333333333333333327
333333333333333333328
333333333333333333329
333333333333333333330
333333333333333333331
333333333333333333332
333333333333333333333
333333333333333333334
333333333333333333335
333333333333333333336
333333333333333333337
333333333333333333338
333333333333333333339
3333333333333333333310
3333333333333333333311
3333333333333333333312
3333333333333333333313
3333333333333333333314
3333333333333333333315
3333333333333333333316
3333333333333333333317
3333333333333333333318
3333333333333333333319
3333333333333333333320
3333333333333333333321
3333333333333333333322
3333333333333333333323
3333333333333333333324
3333333333333333333325
3333333333333333333326
3333333333333333333327
3333333333333333333328
3333333333333333333329
3333333333333333333330
3333333333333333333331
3333333333333333333332
3333333333333333333333
3333333333333333333334
3333333333333333333335
3333333333333333333336
3333333333333333333337
3333333333333333333338
3333333333333333333339
33333333333333333333310
33333333333333333333311
33333333333333333333312
33333333333333333333313
33333333333333333333314
33333333333333333333315
33333333333333333333316
33333333333333333333317
33333333333333333333318
33333333333333333333319
33333333333333333333320
33333333333333333333321
33333333333333333333322
33333333333333333333323
33333333333333333333324
33333333333333333333325
33333333333333333333326
33333333333333333333327
33333333333333333333328
33333333333333333333329
33333333333333333333330
33333333333333333333331
33333333333333333333332
33333333333333333333333
33333333333333333333334
33333333333333333333335
33333333333333333333336
33333333333333333333337
33333333333333333333338
33333333333333333333339
333333333333333333333310
333333333333333333333311
333333333333333333333312
333333333333333333333313<br

- An additional register may be incremented by the length of the converted field based on API parameters.
- Standard Linkage may be generated for conversion routines based on API parameters.
- 5 • Conversion Error exits may be specified to handle enumerated conversion error conditions based on API parameters.
- Character Code Set translation conversion code can be generated based on API parameters (i.e., ASCII character fields can be translated to EBCDIC character fields).
- Conversion routines can be generated to utilize the latest instructions supported by the level of the operating system for which the code is being generated.

In one embodiment, system 20 dynamically generates code by building code chunks in storage accessible by calling application 10 based on various settings in the 15 API control block. Generating the code involves the following steps, as discussed in conjunction with the flowcharts:

1. Obtain storage for the code.
2. Identify code templates needed.
3. Move code templates.
4. Modify code templates.

5. Return executable code to calling application.

Further, in one embodiment system 20 can optionally, based on the API specification, generate program debugging instrumentation for the dynamically generated code. This instrumentation can include an optional dynamically allocated output file containing, for each field conversion: a report of the API options used for each dynamically generated routine that can be used to insure correctness of field attributes and general processing options; and a disassembled listing of the dynamically generated routine provided by an internal disassembler within system 20 that can be used to identify conversion code inaccuracies and areas of further optimization, and to help resolve generated code failures.

15 Figs. 5a and 5b illustrate a general example of dynamic code building that is used in one embodiment of the present invention.

Figs. 6a - 6h illustrate a specific example of a dynamic code generation routine that performs CHARACTER to CHARACTER conversions.

15 Several embodiments of the present invention are specifically illustrated and/or described herein. However, it will be appreciated that modifications and variations of the present invention are covered by the above teachings and within the purview of the appended claims without departing from the spirit and intended scope of the invention.